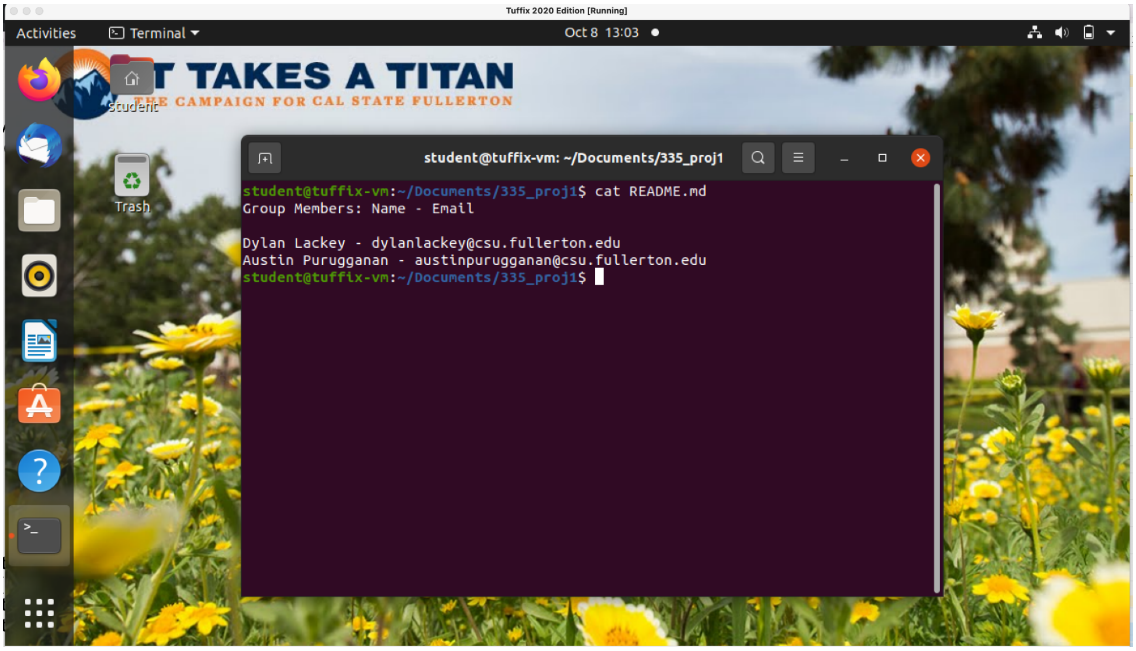
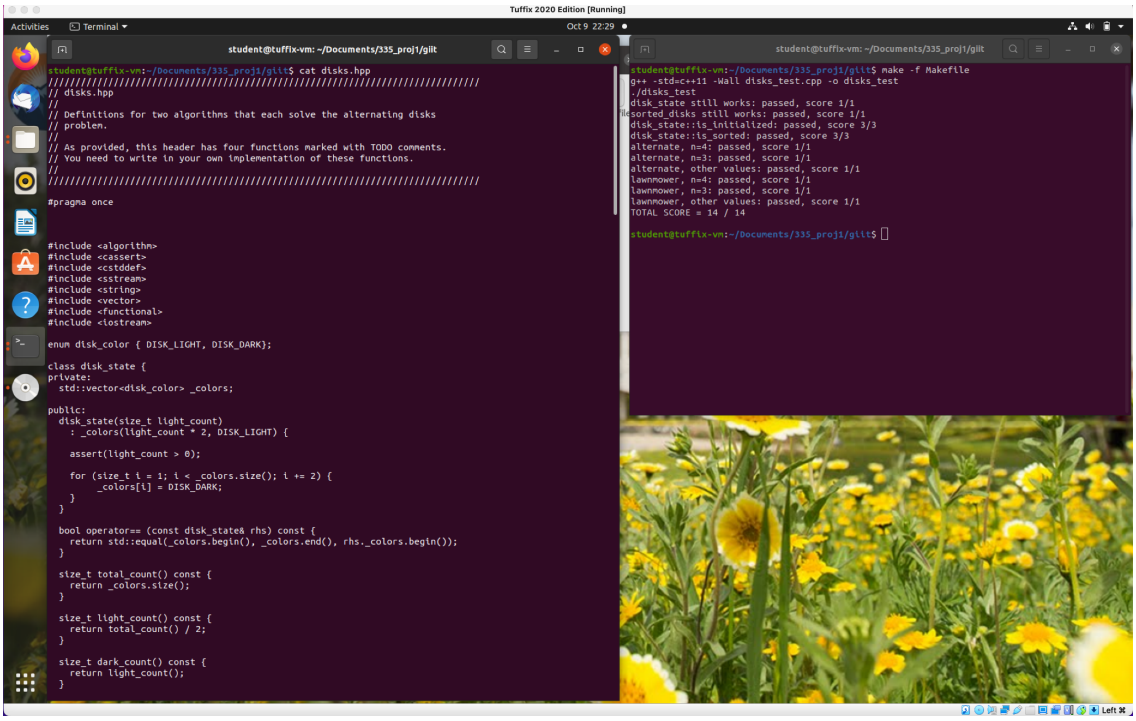
CPSC 445 Project One: Report

Dylan Lackey - dylanlackey@csu.fullerton.edu
Austin Purugganan - austinpurugganan@csu.fullerton.edu

Tuffix and ReadMe



Code and compilation



```
bool is_sorted() const
                {
                for (size_t i = 0; i < total_count(); i++)
                        {
                        if (i < total_count() / 2)
                                {
                                if (_colors[i] == DISK_DARK)
                                        {
                                        return false;
                                        }
                                }
                        else
                                 {
                                 if (_colors[i] == DISK_LIGHT)
                                            return false;
                                 }
                   }
                 return 1;
           }
```

```
// Algorithm that sorts disks using the lawnmower algorithm.
sorted_disks sort_lawnmower(const disk_state& before) {
           int swaps = 0;
          disk_state state = before;

          for (long unsigned int i = 0; i < state.total_count() / 2; i++){
          long unsigned int index = 0;
          while (index + 1 < state.total_count()){
                        if(state.get(index) != state.get(index + 1)){
                                  if(state.get(index) == DISK_DARK && state.get(index+1) == DISK_LIGHT){
                                  state.swap(index);
                                  swaps++;
                                  }//endif
                        }//endif
       index++;
          }//endwhile

          while(index > 0){
                        if(state.get(index-1) != state.get(index)){
                                  if(state.get(index - 1) == DISK_DARK && state.get(index) == DISK_LIGHT){
                                  state.swap(index - 1);
                                  swaps++;
                                  }//endif
                        }//endif
          index--;
          }//endwhile
          }//endfor

 return sorted_disks(disk_state(state), swaps);
       }
```

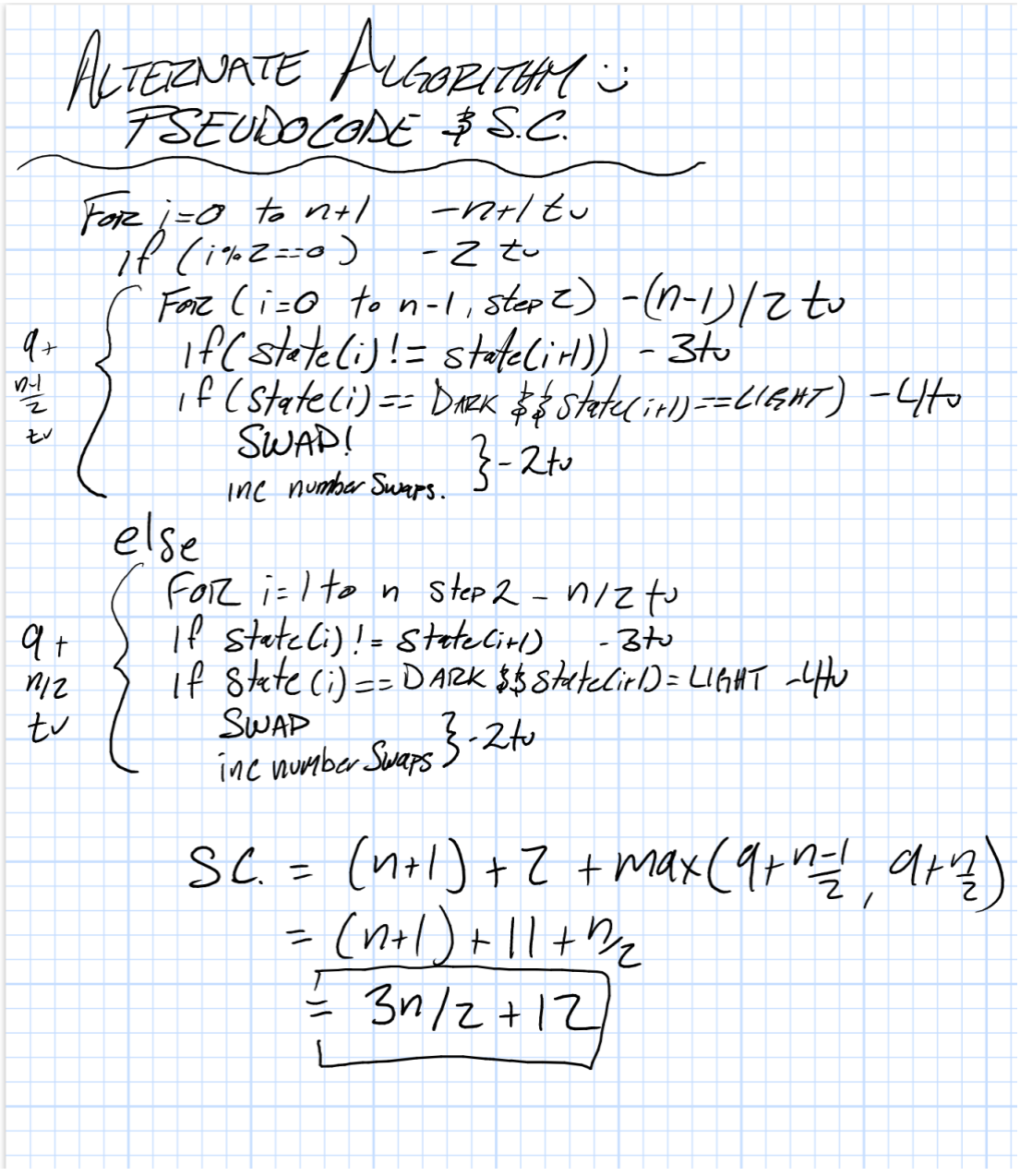
```
// Algorithm that sorts disks using the alternate algorithm.
 sorted_disks sort_alternate(const disk_state& before)
 {
 int numOfSwap = 0;                                                              //record # of step swap
 disk_state state = before;

 for (long unsigned int i = 0; i < state.total_count() + 1; i++)
        {
        if (i % 2 == 0)
                {
                for (long unsigned int index = 0; index < state.total_count() - 1; index=index+2)
                        {
                        if (state.get(index) != state.get(index + 1))
                               {
                                if (state.get(index) == DISK_DARK && state.get(index + 1) == DISK_LIGHT)
                                        {
                                        state.swap(index);
                                        numOfSwap++;
                                        }

                               }
                        }
                }
        else
                {
                for (long unsigned int index = 1; index < state.total_count() - 2; index=index+2)
                        {
                               if (state.get(index) != state.get(index + 1))
                                {
                                if (state.get(index) == DISK_DARK && state.get(index+1) == DISK_LIGHT)
                                        {
                                          state.swap(index);
                                          numOfSwap++;
                                        }
                                }
                        }
                }
        }
   return sorted_disks(disk_state(state), numOfSwap);
 }
```

Pseudocodes and step counts

Alternating Algorithm



Lawnmower Algorithm
(pseudo code & SC)

```
j=0; //index
counter=0; //swap counter
for i=0 to n/2, i++ -> n/2 times
  {while(j+1<n)  -> n-1 times
    {if(state[j] != state(j+1)   2tu
       if(state[j]==dark && state[j+1]==light) 3tu
         {swap[j];   1tu
           counter++;} 1tu
     j++  1tu
    }
  while(j>0)
    {if(state[j-1] != state[j]  2tu
       if(state[j-1]==dark && state[j]==light) 3tu
         { swap[j-1];  1tu
           counter++; } 1tu
     j--  1tu
    }
```

$$SC = (n/2)[(n-1)(8)+(\log_2 n+1)(8)]$$

$$= 4n[n-1+\log_2 n+1]$$

$$= \boxed{4n(\log_2 n + n)}$$

---

https://github.com/apurugganan0/project-lawnmover